**Опитувальник з теми Java, ООП, ООД, SOLID, DRY, KISS, YAGNI**

**Java та ООП**

1. **Що таке об'єктно-орієнтоване програмування (ООП)?**
   * a) Методика розробки програмного забезпечення, яка використовує модулі.
   * b) Парадигма програмування, яка використовує об'єкти та класи.
   * c) Спосіб управління проектами.
   * d) Методика розробки баз даних.
2. **Назвіть чотири основні принципи ООП.**
   * a) Інкапсуляція, Наслідування, Поліморфізм, Агрегація.
   * b) Інкапсуляція, Наслідування, Поліморфізм, Абстракція.
   * c) Інкапсуляція, Композиція, Поліморфізм, Наслідування.
   * d) Інкапсуляція, Інтерфейси, Поліморфізм, Наслідування.
3. **Що таке інкапсуляція?**
   * a) Механізм приховування деталей реалізації об'єкта.
   * b) Спосіб створення об'єктів.
   * c) Процес створення нових класів на основі існуючих.
   * d) Механізм перевантаження методів.
4. **Що таке поліморфізм?**
   * a) Здатність об'єкта приймати різні форми.
   * b) Здатність методу викликати себе рекурсивно.
   * c) Здатність програми виконуватися на різних платформах.
   * d) Процес створення об'єктів з існуючих класів.

**ООД та SOLID**

1. **Що таке об'єктно-орієнтоване проектування (ООД)?**
   * a) Методика розробки баз даних.
   * b) Підхід до проектування систем, який використовує концепції ООП.
   * c) Спосіб управління проектами.
   * d) Методика розробки інтерфейсів.
2. **Що означає принцип Single Responsibility (SRP) з SOLID?**
   * a) Клас повинен мати одну відповідальність.
   * b) Клас повинен мати кілька відповідальностей.
   * c) Клас не повинен мати жодної відповідальності.
   * d) Клас може мати одну або кілька відповідальностей.
3. **Що означає принцип Open/Closed (OCP) з SOLID?**
   * a) Класи повинні бути відкритими для модифікацій, але закритими для розширення.
   * b) Класи повинні бути закритими для модифікацій, але відкритими для розширення.
   * c) Класи повинні бути закритими для модифікацій і розширення.
   * d) Класи повинні бути відкритими для модифікацій і розширення.
4. **Що означає принцип Liskov Substitution (LSP) з SOLID?**
   * a) Підкласи повинні бути взаємозамінними з базовими класами.
   * b) Підкласи повинні мати більше методів, ніж базові класи.
   * c) Підкласи повинні мати менше методів, ніж базові класи.
   * d) Підкласи повинні бути взаємозамінними з іншими підкласами.
5. **Що означає принцип Interface Segregation (ISP) з SOLID?**
   * a) Інтерфейси повинні бути великими і об'єднувати кілька функцій.
   * b) Інтерфейси повинні бути маленькими і специфічними для окремих дій.
   * c) Інтерфейси не повинні використовуватися в програмах.
   * d) Інтерфейси повинні містити тільки один метод.
6. **Що означає принцип Dependency Inversion (DIP) з SOLID?**
   * a) Модулі верхнього рівня не повинні залежати від модулів нижнього рівня.
   * b) Модулі нижнього рівня повинні залежати від модулів верхнього рівня.
   * c) Обидва типи модулів можуть залежати один від одного.
   * d) Модулі повинні бути ізольованими і не залежати один від одного.

**DRY, KISS, YAGNI**

1. **Що означає принцип DRY?**
   * a) Кожна частина коду повинна бути унікальною.
   * b) Кожна частина коду повинна бути повторюваною.
   * c) Не повторюйте себе.
   * d) Кожен клас повинен мати одну відповідальність.
2. **Що означає принцип KISS?**
   * a) Тримай це простим і зрозумілим.
   * b) Тримай це коротким і простим.
   * c) Тримай це структурованим і специфічним.
   * d) Тримай це складним і специфічним.
3. **Що означає принцип YAGNI?**
   * a) Вам це знадобиться.
   * b) Вам це не знадобиться.
   * c) Використовуйте це у випадку необхідності.
   * d) Використовуйте це завжди.
4. **Яка основна мета принципу DRY?**
   * a) Уникнути дублювання коду.
   * b) Покращити продуктивність.
   * c) Спрощення структури коду.
   * d) Зменшити розмір коду.
5. **Чому принцип KISS важливий у розробці програмного забезпечення?**
   * a) Він дозволяє створювати складні рішення.
   * b) Він забезпечує простоту і легкість у розумінні коду.
   * c) Він збільшує кількість функцій.
   * d) Він зменшує кількість тестів.

**Відповіді:**

1. b
2. b
3. a
4. a
5. b
6. a
7. b
8. a
9. b
10. a
11. c
12. a
13. b
14. a
15. b